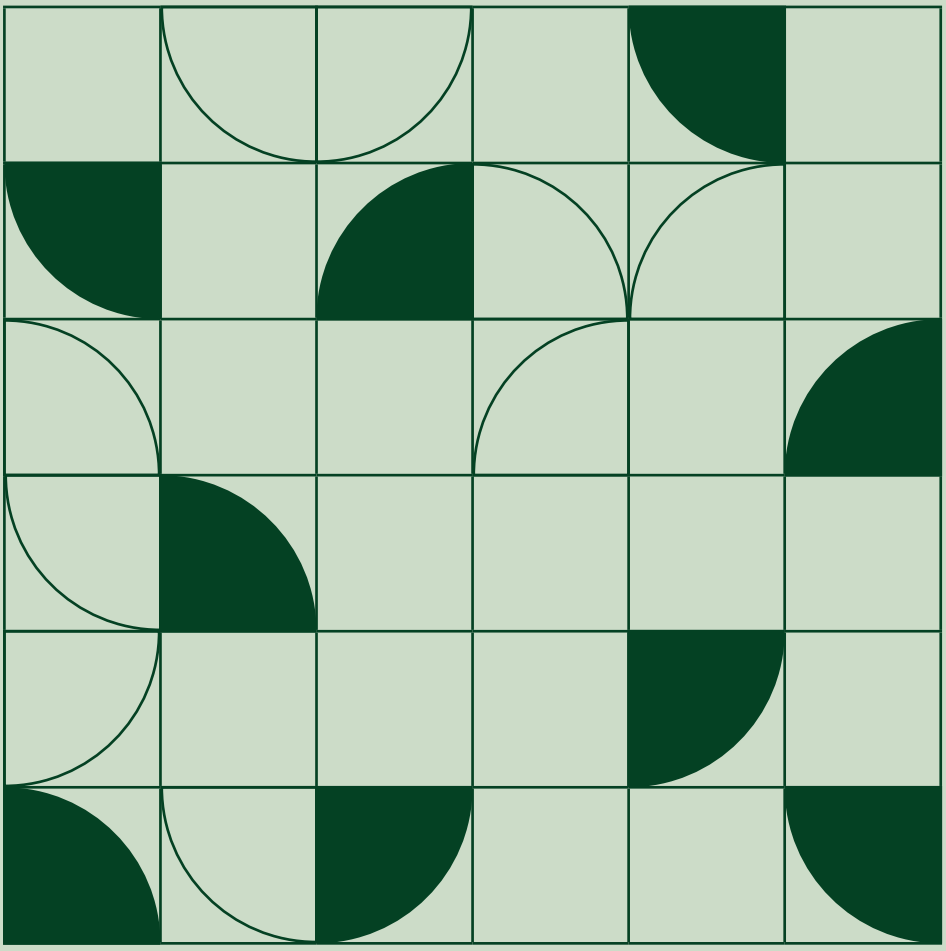
测试执行软件



简介

大多数测试系统都是围绕着“效率”和“成本”这两个概念进行设计。无论是在消费电子产业还是半导体生产领域，测试工程师都非常关心测试系统的独立测试时间和总吞吐量，以及这些参数如何影响资源。当应用程序不断扩展到包含多种测试、各种仪器和多个待测设备(UUT)时，便不可避免地需要监督测试执行软件，以解决成本和效率问题。

测试执行软件通常由客户在内部自行开发，或者作为商用现成(COTS)产品购买。在典型的构建还是购买的争论中，测试架构师必须确定编写自定义测试执行程序更合适，还是购买现成解决方案加以集成更具成本效益。在决定自主开发还是购买测试执行软件之前，测试架构师必须了解此类软件的目的和核心功能。本指南总结了测试执行软件的主要功能，并介绍了这一技术的实际应用场景。

背景

测试执行软件可实现大型测试系统的自动化和精简化。该解决方案位于软件堆栈的最上层，整合了各测试级别的通用功能，如测试执行、结果收集和报告生成。测试执行软件的功能特性并非专属于特定UUT，因此各种应用均可将其作为框架使用。这意味着开发人员在使用LabVIEW图形化语言、C、.NET或其他语言编写测试代码的时候可以专注于测试具体的设备，而所有UUT的常用功能都由最上层测试执行软件维护。总而言之，测试执行软件从开发、成本和维护的角度，有效地定义了这些通用功能。

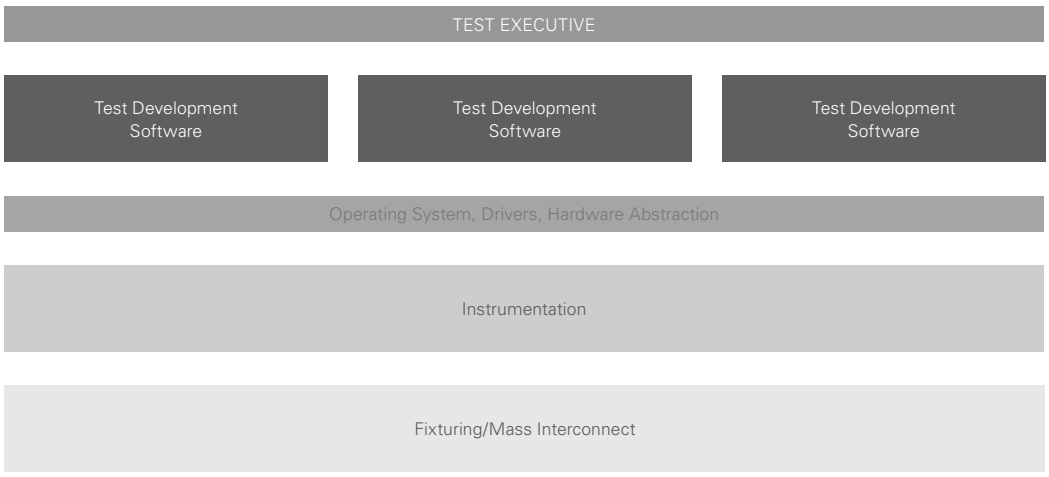


图1 通过在更高抽象级别上完成所有的通用测试任务，测试执行软件将个体测试开发与整体测试系统架构需求分离

测试执行软件的特性

根据公司大小、测试仪具体规模以及被测设备的种类，执行测试软件分为简单版到高级版等不同复杂度。本指南概述了该软件可能包含的常见特性。某些特性对于测试执行软件的所有实现都至关重要，另一些则并非绝对必要。每种特性都预估了完成开发的用时。这些预估数据基于数百个自动化测试客户的经验，具体可见《测试执行软件-构建还是购买？基于NI TestStand的财务分析比较》。

测试序列开发环境

测试执行软件提供用于构建测试序列的开发环境。该特性可为整个执行过程提供开发接口，既是不可或缺的基础，同时也十分复杂。序列架构包括运用分支或循环逻辑的能力、导入测试限制的方法以及独立测试代码的规范和整理。与测试代码的交互需要具有使用各种编译格式的灵活性，例如DLL、VI和脚本，以及与不同开发环境集成的能力。测试执行软件也可以使用源代码控制程序提供的测试代码。

使用自定义测试执行软件构建测试序列开发环境需要约100个人日，而购买商用解决方案可立即获得现成的环境。自行开发解决方案将耗费大量开发时间，因为开发环境涉及的功能范围十分广泛。然而，序列开发环境是序列架构体验的基础，不可忽视。

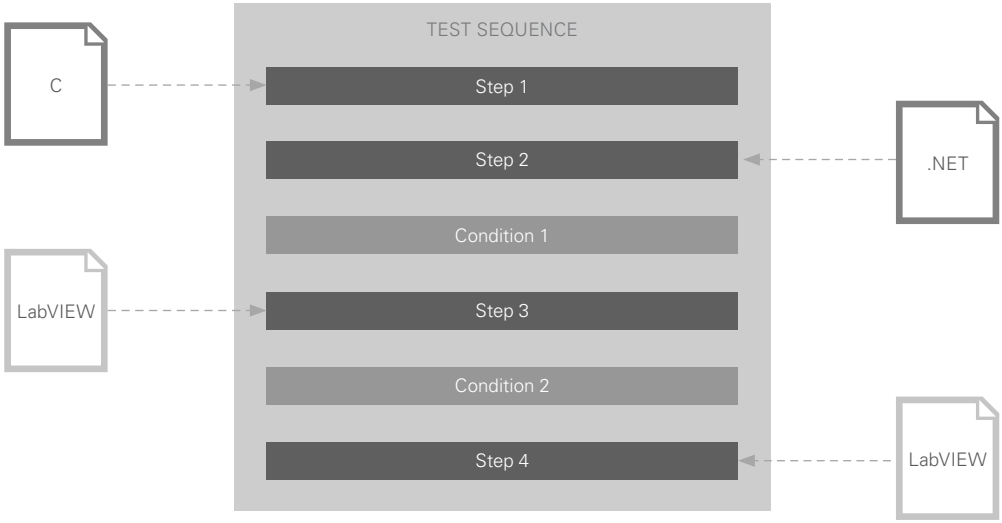


图2 高效的序列开发环境可帮助测试工程师开发和调试复杂的序列，并调用到现有测试代码中

自定义操作界面

操作界面是操作员用来与测试系统进行交互的显示器。操作界面通常允许操作员选择关键输入参数，例如UUT标识符、待执行的测试序列或报告路径。其中还包含一个“运行”或“开始”按钮，用以控制执行过程。如今，许多大型测试系统都需要专业GUI，该GUI因应用或公司而异，由开发人员所选的编程语言编写而成。除了自定义功能，高度功能性的界面还具有加载、显示和运行测试序列的能力，同时配有交互式用户提示、执行进度指示器、测试数据可视化和本地化功能。

开发自定义操作界面可能需要8到32个人日。COTS解决方案提供现成的库和UI控件，可以缩短该时间。无论是自行构建还是购买测试执行解决方案，自定义操作界面的开发都需要投入相当一部分时间。认为操作界面对其系统无足轻重的测试工程师可能会指示操作员直接在开发环境中工作。

序列执行引擎

测试执行软件的核心配置是序列执行引擎。序列执行引擎负责UUT评估所需的所有操作，包括调用独立测试代码、创建测试间的执行流程以及管理测试间的数据。无论是在开发环境中、通过自定义操作界面，还是在已部署的测试仪上，序列执行引擎均可执行给定的测试序列。

自行开发一款序列执行引擎需要至少15个人日的工作量。但它对于所有测试执行软件来说不可或缺。

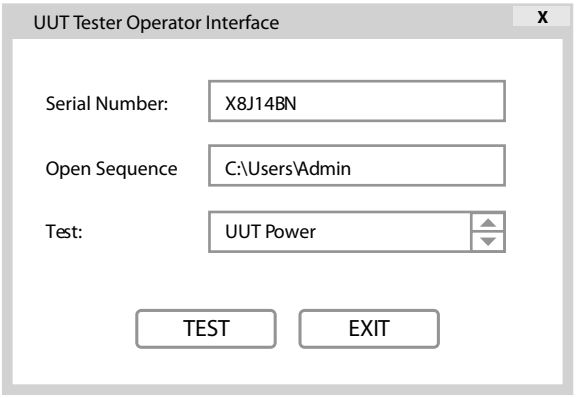


图3 客户操作界面能够识别给定测试序列的唯一UUT、公司、应用程序、测试和操作员角色

结果报告

测试执行软件扮演着抽象的角色，负责整合个体测试数据、将数据临时储存至内存，以及发布综合测试结果。报告采取多种格式，包括XML、text、HTML和ATML格式。数据也可以在执行后存储到数据库。通过可扩展的报告选项，测试执行软件提供多样化的报告格式。结果报告对于许多测试系统来说必不可少。

从零开发结果收集功能和报告生成器大约需要15个人日，具体取决于所需的报告。COTS解决方案内置了报告生成器，可以在1个人日或更短时间内通过自定义结果报告满足应用需求。

用户管理

将不同角色在测试执行软件级别的责任区分开来很有必要。用户管理工具有效划分了总测试架构师、编写和调试测试代码的个体测试开发人员以及运行测试的操作员或生产经理之间的责任。某些面向特定用户的功能甚至可能受到密码保护，以防测试序列出现误用。

使用自定义测试执行软件开发用户管理系统大约需要五个人日。尽管并非测试执行软件的必需工具，但用户管理工具无需开发人员投入过多精力即可实现，而且可以简化测试执行软件职责的执行。

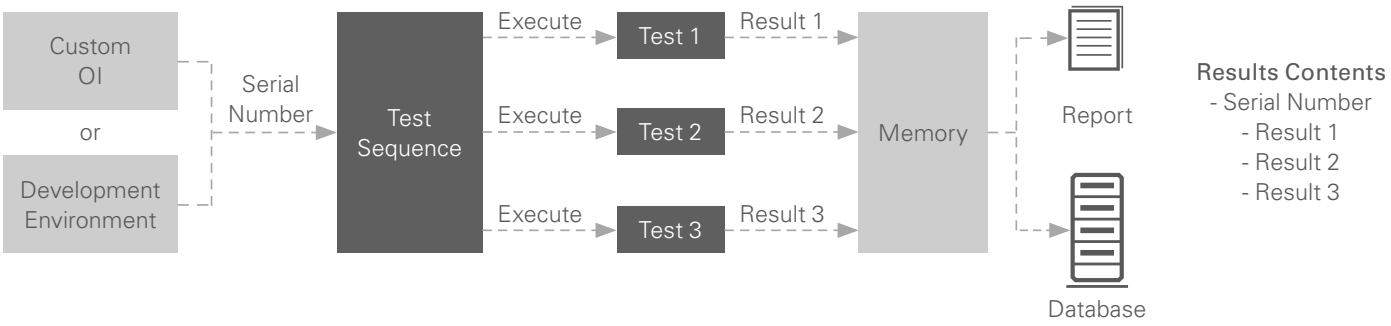


图 4 测试执行软件在测试系统中的部分职责是整合执行结果并将其发布到报告或数据库中

权限	架构师	开发人员	操作员
编辑	√	-	-
保存	√	-	-
部署	√	√	-
循环	√	√	-
运行	√	√	√
退出	√	√	√

用户	级别
Mark	操作员
Larry	操作员
Julie	开发人员
Scott	开发人员
Lauren	架构师

表 1 与Windows文件权限类似，用户管理器可将不同角色在测试执行软件级别的责任区分开来

并行测试功能

并行测试涉及同时测试多个设备，与此同时仍保持代码模块性能、结果收集和UUT跟踪功能正常运行。并行测试方法的范围涵盖流水线执行（测试顺序保持不变，但测试执行软件可以同时跨多个测试座测试）、动态优化、批处理以及其他复杂的执行方式。

对于测试执行软件开发人员来说，开发并行测试功能通常极为耗时，从头开始可能需要100个人日。尽管并行测试需要较长的开发时间，但在大型测试系统中，通过扩展执行以减轻吞吐量需求的能力至关重要。许多公司在首次部署测试执行软件时并未考虑并行测试功能，后来才意识到此功能不可或缺，但为时已晚。

单元/设备跟踪和序列号扫描

同时测试多个UUT设备时，有必要对每个被测设备进行唯一的标识和跟踪。这些信息可以与测试结果一起存储，以进行特定单元或批次分析，或者用于在出现问题时查明错误来源。设备跟踪的方式包括由操作员在键盘上手动输入，以及使用全自动扫描仪读取条形码以加载UUT信息等。

完全自主开发此功能大约需要5个人日，使用COTS解决方案进行定制开发则需要大约1个人日。并非所有测试系统都需要UUT跟踪。但此功能对于半导体或消费电子等需要大容量、高吞吐量测试的行业来说非常有用。

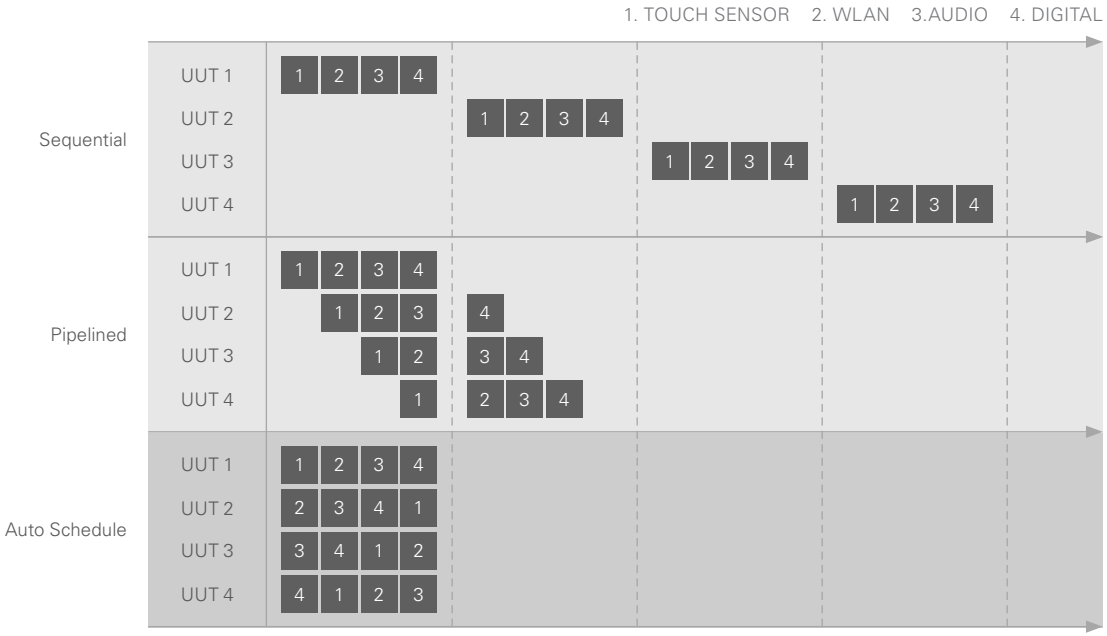


图 5 并行测试功能无需重建测试执行软件架构即可提升系统吞吐量

测试部署工具

大多数大型测试系统并不会采用孤立架构，而是多个测试站点或整个生产车间的解决方案。测试执行软件在系统部署中发挥着关键作用，它提供一种机制或实用程序，可将整个软件堆栈打包到内置的可分发单元中。测试系统可通过多种方式分发，架构师可能更希望部署测试系统的镜像或者部署包含所有必要依赖项和运行时间的全功能安装程序。有关此主题的更多信息，请参阅《测试系统构建基础知识》系列技术白皮书的“软件部署”部分。

部署是一项艰巨的任务，如果从头做起，开发团队可能需要多达20人日的工作量。即便使用商用测试执行软件提供的现成部署实用程序，可能仍然需要三个人日才能成功部署。考虑到此工具适用于多个测试站点，通常有必要在测试执行解决方案中配备此工具。

维护

就像大型测试系统中的任何其他组件一样，测试执行软件必须得到适当的维护，以确保经久耐用。维护内容包括进行扩展以涵盖新测试、维持软件或操作系统升级的兼容性，以及修复检测到的所有错误。测试执行解决方案的维护甚至延伸到文档领域，文档是操作员、开发人员和架构师在与测试执行人员合作时所赖的重要资源。

尽管无法得知具体测试仪的需求，但在每年开发自定义测试执行软件的初期，有15%的时间需要用于维护。总开发时间包括生成充足文档所需的时间（预计为20天）。测试执行软件支持的时间间隔因具体情况而异，有可能大幅增加上述开发的成本估算值。尽管如此，任何测试执行解决方案都不应轻视维护工作。

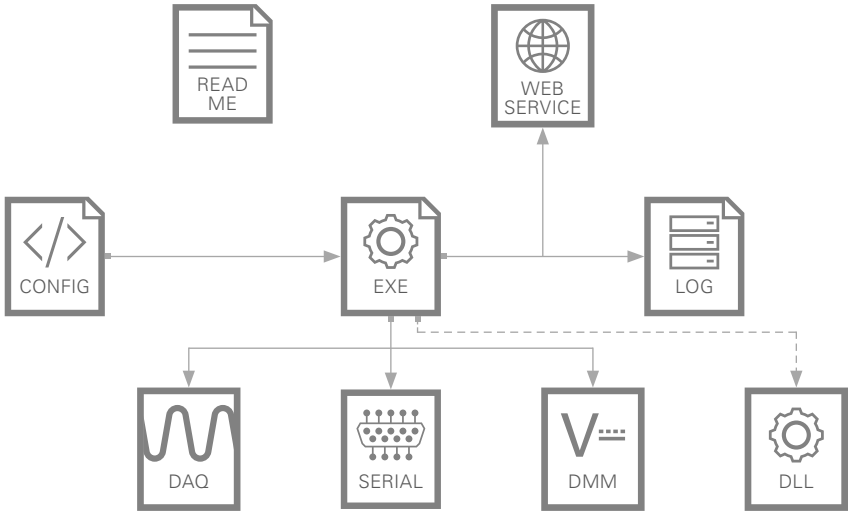


图 6 部署涉及使用部署工具或编译服务器打包测试系统的所有必要组件，然后将打包后的内容分发到目标测试站

实际场景1

Jonathan在一家小型公司的设计实验室担任测试工程师，该公司主要生产低成本消费电子产品。每个设备由一个远程控制器控制，Jonathan负责编写测试代码，在设备投入生产之前检验远程控制器和设备之间的发射器-接收器通信情况。随着公司近来规模不断扩大，Jonathan对于每台设备的测试时间越来越有限。他希望让系统自动执行现有验证代码，以便留出更多时间为新设备编写代码。因此，他决定使用测试执行软件来执行测试代码序列。

Jonathan对测试执行软件的需求如下表所示。

功能特点	实现方式
测试序列开发环境	Jonathan需要一个开发环境来构建序列。他一直以来使用的测试代码已经相当模块化，所以他只需要在这个环境中调用和循环测试代码。
自定义操作界面	Jonathan希望能够以最少的交互执行一组测试代码。他希望只需要通过指定几个相关参数就能够识别设备、所需测试和报告路径。但是，鉴于他是最终操作员，是否拥有一个与开发环境分离的接口并不重要。
序列执行引擎	这是Jonathan对测试执行解决方案的核心需求。每个测试由几个必须按序执行的独立LabVIEW VI组成。
结果报告	现有测试代码会提示用户为给定原型生成新的技术数据管理流（TDMS）文件。随后的每个VI都将最少的测试结果存入该文件。测试执行软件只需自动创建该TDMS文件，然后执行其余的测试代码，即可按约定生成结果。
用户管理	用户管理并非优先事项，因为该测试执行软件由Jonathan本人设计、开发和操作。
并行测试功能	Jonathan每次只测试一个设备，因此UUT的数量不是问题。
单元/设备跟踪和序列号扫描	测试是在设计原型上完成的，因此Jonathan无需跟踪分配的序列号，而是通过操作员在运行时输入的唯一名称跟踪每个UUT。
测试部署工具	Jonathan不打算将此代码部署到其他测试仪。他的测试台为设计实验室所独有，与制造设施互相独立。
维护	这个项目完全属于Jonathan。他将使用和维护所有选中的测试执行软件。他不打算通过文档记录自己的工作，因为他将是唯一一名操作和使用此测试执行软件的用户。

表2 Jonathan对测试执行软件的需求主要集中于现有验证代码的简单自动化

Jonathan决定自主开发一款测试执行软件。他没有复杂的序列执行或报告需求，也不打算将此系统部署到其他用户或测试站。如果购买了商用测试执行软件，他可能之后无法取得投资回报，因为其中的大部分功能都派不上用场。相比之下，他只需依靠自己的软件知识和先前所购买的应用软件，即可在短短10个人日内开发出满足需求的序列执行器。

Jonathan在LabVIEW软件中开发了他的测试执行软件。他的设计拥有简单界面，让操作员能够调用一组预定的测试步骤并选择TDMS日志路径。在为新的原型引入额外测试时，Jonathan偶尔会对序列执行器做一些小改动。总的来说，得益于这款序列执行器，设计实验室的生产力得到了显著提升。

在这种情况下，自行开发测试执行软件是满足Jonathan标准的最佳选择。通常来讲，自主开发测试应用软件是需要序列生成或完全自动化时所采取的第一步，相比于生产环境的需求，此方法可能更适合测试台应用。

功能特点	实现方式
测试序列开发环境	高效的开发环境必不可少，这个环境必须支持测试执行软件的主要功能特性，且必须支持LabVIEW、.NET和Python代码的序列执行。
自定义操作界面	Dave的最终需求是一个为公司量身定制的操作界面。他还希望删除“运行”按钮之外的多数功能。
序列执行引擎	显而易见，此功能是为了满足该系统的吞吐量需求。
结果报告	目前，每次测试都会将数据单独记录到SQL数据库中。因此，测试执行软件需要整合收集的结果、将结果汇总传送到数据库并使用序列号进行标识。
用户管理	大多数与测试仪的交互由生产级别的操作员进行。Dave更希望用户管理工具或可自定义界面能够从操作员的界面中删除开发权限。
并行测试功能	只要测试仪吞吐量与生产吞吐量相匹配，Dave就不需要一次性测试多个UUT。
单元/设备跟踪和序列号扫描	序列号用于标识每个组件和组装的UUT。条形码扫描仪用于跟踪此类信息。测试执行软件必须能够在其执行的不同测试之间传输此类信息。
测试部署工具	Dave需要将最终产品部署到另外10个测试仪中。
维护	测试工程部门将维护测试执行软件，可能是在内部解决方案维护全部功能，也可能在COTS软件中按需维护。

表3 | Dave对测试执行软件的评估主要基于功能测试仪的潜在吞吐量需求

如果...

- 几个月后，设计实验室聘请了新的测试工程师来协助测试。该工程师该如何学习序列执行工具的操作方法，或有效解决任何出现的错误或漏洞？
- Jonathan调到另一个部门，或者离开公司。更新和维护序列执行工具所需的知识库应当如何维护？
- 序列执行工具有必要对整个原型进行功能评估。如何将不同工程师用其他语言编写的附加测试代码、不同的编程范式和报告技术整合到一起？
- 测试执行软件被移植到生产环境中，以确保测试的一致性。这些软件能否因地制宜，满足此类需求？

实际场景2

Dave的公司正在设计一款新的功能测试仪，将在生产线末端使用。目前，运行UUT测试的方式是手动执行一系列已有的零散代码片段。这一过程极大限制了生产线的吞吐量，Dave希望使用测试执行软件将其自动化。该公司没有针对测试执行软件的统一标准，每个小组通常会从少部分商用解决方案和众多自定义解决方案中选择所需软件。

Dave对测试仪的基本要求如表3所示。

为了做出决定，Dave还从财务角度对测试仪进行了权衡。他估计新的测试仪将是一个大型的高性能PXI机箱和嵌入式控制器的组合。考虑到评估UUT所需测试的性质，机箱将包含从数据采集卡和模块化仪器（如数字化仪和任意波形发生器）到射频测试设备的多个模块。无论采用何种测试执行软件，每台测试仪的成本都将在10万美元左右。

在评估软件堆栈时，Dave注意到购买COTS解决方案会增加项目成本。购买测试执行软件的开发许可证需要几千美元，并且每增加一台测试仪，都要另外花费500美元获得测试仪的部署许可证。

Dave相信他可以通过使用Python构建自定义解决方案来节省测试执行软件的成本。该语言是开源的，并且开发环境可免费使用，他认为这两项优势足以抵消在内部开发测试执行软件所需的额外开发时间。

测试工程团队精通Python，最终在规定的时间范围内提供了顺序序列执行引擎、数据库连接和现有测试代码复用等核心功能。测试执行软件现已成功部署到生产线。测试工程师偶尔会被要求修复一台或多台测试仪的错误。

如果...

- 生产线的生产需求增加，以至于现有的测试执行软件无法满足吞吐量需求，此时需要添加并行测试功能。
  - 尝试实现此功能需要多少额外的开发时间？这对自定义测试执行软件和COTS解决方案的成本对比有何影响？
  - 假设由于Python语言在多处理方面的局限性，测试仪吞吐量无法满足需求。Dave的团队需要购买额外的硬件，以便再次利用目前已有的测试执行软件，或是选择另一种全新的测试执行软件。这对自定义测试执行软件和COTS解决方案的成本对比有何进一步影响？
- 因为还有其他优先任务，测试工程团队无法随时为测试执行软件提供服务或升级。
  - 当出现此类需求而团队无法及时提供帮助时，生产会受到怎样的影响？此类停机对系统维护成本有何影响？
  - 测试工程团队花费在维护测试仪上的时间是如何量化的？此因素对系统维护成本有何影响？

实际场景3

Karen在一家设计和生产小型医疗设备的公司工作。该公司为每种产品设立了单独的全自动生产线。虽然每个小组都采用测试执行软件进行最上层的系统管理，但该公司尚未实现测试执行软件的标准化。最近，该公司的新任测试经理上任，有意对测试执行软件进行标准化。Karen的任务是选择商用解决方案、现有内部产品或进行新的开发来实现测试执行软件的标准化。

Karen为负责每种产品的各个小组编写了以下要求列表。

功能特点	实现方式
测试序列开发环境	测试开发人员需要一个灵活的开发环境，特别是可以与他们的LabVIEW和VB.NET代码交互。Tortoise SVN用于源代码控制，开发环境需要与此工具集成。
自定义操作界面	测试经理希望根据正在开发或测试的产品自定义操作界面。操作员表示他们希望在监控测试仪时能有一个进度指示器实时更新测试状态。
序列执行引擎	这是所有测试仪必需的功能。
结果报告	所有生产系统都必须符合公司的HTML报告标准。
用户管理	测试工程团队由少数系统架构师和大量测试开发人员组成。测试经理想要将这两个角色的职责区分开来。
并行测试功能	在对组装好的单元进行功能测试时，生产线一次评估一个UUT。然而，板卡级测试应进行优化，以提升执行速度。为了满足所有测试仪的需求，需要添加并行测试功能。
单元/设备跟踪和序列号扫描	公司每个产品和板卡的UUT信息通过操作员输入进行跟踪。
测试部署工具	该公司拥有专门编写测试代码的测试工程师团队。该团队必须能够将所在实验室的开发环境部署到所需的生产环境。目前，部署过程需手动完成。
维护	作为标准化工作的一部分，测试经理需要一个正式的维护计划。该计划需要考虑到公司在今年晚些时候当前系统生命周期结束时进行的操作系统迁移。

表4 | Karen对测试执行解决方案的兴趣源于对不同测试仪进行标准化的需求

根据此标准，Karen排除了所有现有的内部解决方案。这些解决方案大多只能用于单个测试仪，其架构缺少一致性，无法扩展到其他生产线，特别是在序列执行需求、操作界面定制和有效部署实践方面。此外，过往事实证明，当软件出现问题或对设备进行修改时，很难追踪到负责该测试执行软件的测试工程师。

因此，Karen向她的经理提议购买商用解决方案。该测试执行软件由知名供应商制造，同一供应商生产的其他硬件和软件工具已广泛应用于测试仪中。该测试管理软件支持“即开即用”，可以满足测试仪的序列执行需求，并采用所要求的特定报告格式。该测试执行软件还包括一组可满足其他测试仪需求的工具，包括用户管理工具和部署实用程序。由于商业供应商提供维护服务，Karen的经理不必担心日后操作系统迁移时会产生不兼容的问题。

Karen的公司最终做出了成功的决策。总体而言，测试执行软件提供了一个适用于不同生产线的灵活框架。购买测试执行软件的统一标准为公司带来了额外的好处。供应商提供培训服务，可帮助测试工程师熟悉新软件。测试执行软件的购买合同还包括维护服务，由供应商同意提供常规补丁和升级。公司还可以获得技术支持资源，有助于解决测试序列的问题。

这款商用解决方案至今仍然是Karen公司的标准。当测试工程师因升职、退休或自然减员等原因而需要换人时，测试经理可以聘请具有测试执行软件操作经验的人员。该公司成功地从一个过时的操作系统迁移到两个完整版本，同时仍保留所选的测试执行软件。随着新产品的开发，可扩展架构可以持续满足生产需求。

结论

无论公司规模、行业或个人测试标准如何，都有必要部署测试执行软件来实现最上层的系统管理。这意味着将引入一定程度的抽象，将系统的常用功能与测试代码的特定功能分离。在构建最终解决方案前，必须对测试执行软件的需求进行完整的评估。许多测试工程师都在自主开发和购买测试执行软件犹豫不定，需要从成本、功能和维护的角度仔细考量每个解决方案的优势，进行取舍。

附加信息

TestStand是一款行业标准的测试管理软件，可帮助测试和验证工程师更快地构建和部署自动化测试系统。TestStand包含可立即运行的测试序列引擎，支持多种测试代码语言、灵活的结果报告和并行/多线程测试。

TestStand不仅包含了许多开箱即用的功能，而且也具有高度可扩展性。因此，全球数以万计的用户选择TestStand来构建和部署自定义自动化测试系统。NI提供培训和认证计划，每年培训和认证超过1000名的TestStand用户。

了解更多关于TestStand的信息